Step by Step Guide to Deploy Static ReactJS Web Application on S3 and CloudFront

Create React App

```
npx create-react-app portfolio
```

Change you director to where you created the app and start you application

```
cd portfolio
npm start
```

Deploy React App to AWS S3 Bucket

- Create S3 bucket
- Enable Static web hosting on the S3 and use **index.html** for the your app entry default document
- Build your ReactJS application

```
npm run build
```

Setup CI/CD Pipeline for React App using GitHub Actions

- Create an empty public GitHub repository and call it portfolio
- Inside application folder start github tracking as follows

```
git init
```

- Push an existing repository from the command line – run the following command to link the repository with the local react app you are working with

```
git remote add origin git@github.com:aetana/portfolio.git
git branch -M main
git push -u origin main
```

- Create GitHub Actions workflow – the following folder structure in your app folder

```
mkdir .github/workflows/main.yml
```

- Define all the necessary setting for
  - dependencies,
  - run some unit tests,
  - upload files to S3 and
  - invalidate the cache

Here is my example yml file

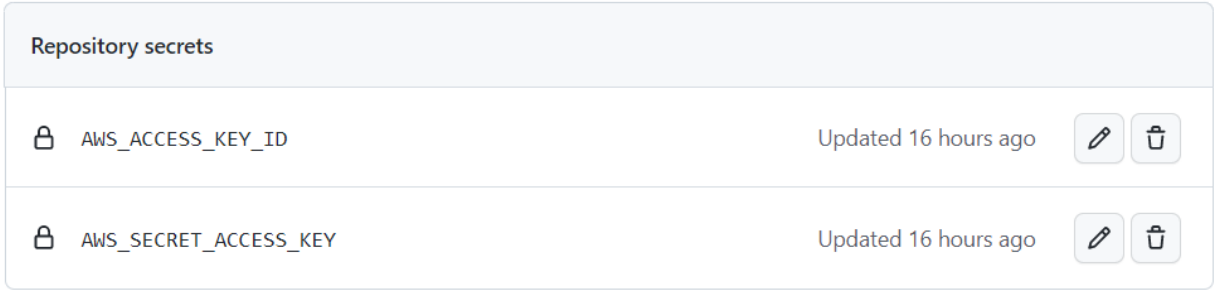
```yaml
name: Deploy ReactJS App to S3
on:
  push:
    branches: [ main ]
jobs:
  build-and-deploy:
    runs-on: ubuntu-latest
    env:
      BUCKET: amanueletana.com
      DIST: build
      REGION: us-east-1
      DIST_ID: E25NXMFBB7832M

    steps:
    - name: Checkout code
      uses: actions/checkout@v2
    - name: Configure AWS Credentials
      uses: aws-actions/configure-aws-credentials@v1
      with:
        aws-access-key-id: ${{ secrets.AWS_ACCESS_KEY_ID }}
        aws-secret-access-key: ${{ secrets.AWS_SECRET_ACCESS_KEY }}
        aws-region: ${{ env.REGION }}
    - name: Set up Node.js environment
      uses: actions/setup-node@v2
      with:
        node-version: 14
    - name: Install dependencies
      run: |
        node --version
        npm ci --production
    - name: Build ReactJS App
      run: npm run build
    - name: Copy files to the production website with the AWS CLI
      run: |
        aws s3 sync --delete ${{ env.DIST }} s3://${{ env.BUCKET }}
    - name: Copy files to the production website with the AWS CLI
      run: |
        aws cloudfront create-invalidation \
          --distribution-id ${{ env.DIST_ID }} \
          --paths "/*"
```

Create an IAM user and grant appropriate permissions needed to grant access for GitHub Actions to upload files to S3

Let's create **S3WebAccess** IAM policy

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ListObjectsInBucket",
            "Effect": "Allow",
            "Action": "s3:ListBucket",
            "Resource": "arn:aws:s3:::www.amanueletana.com"
        },
        {
            "Sid": "AllObjectActions",
            "Effect": "Allow",
            "Action": "s3:*Object",
            "Resource": "arn:aws:s3:::amanueletana.com/*"
        },
        {
            "Sid": "InvalidateCF",
            "Effect": "Allow",
            "Action": "cloudfront:CreateInvalidation",
            "Resource": "*"
        }
    ]
}
```

- Create github-actions user and attach S3WebAccess policy.
- Go to GitHub repo and create AWS_ACCESS_KEY_ID and AWS_SECRET_ACCESS_KEY secrets.



- Make any change in the source code, commit and push.

```
git add .
git commit -m 'update v2'
git push origin main
```

build-and-deploy

succeeded 1 minute ago in 38s

Search logs

Set up job	3s
Checkout code	1s
Configure AWS Credentials	0s
Set up Node.js environment	0s
Install dependencies	14s
Build ReactJS App	11s
Copy files to the production website with the AWS CLI	3s
Copy files to the production website with the AWS CLI	2s
Post Set up Node.js environment	0s
Post Configure AWS Credentials	0s
Post Checkout code	0s
Complete job	0s